**{Epicture.}**

**Coding Academy | Epitech Toulouse**

**Documentation**

# **TABLE OF CONTENTS**

[GENERAL INFORMATIONS 3](#_Toc39504818)

a. General presentation of the project

b. Utilization

c. Functionalities

[PRESENTATION OF THE PROJECT 4](#_Toc39504819)

a. Architecture

b. dependencies

c. automated tests

# **GENERAL INFORMATIONS**

1. **General presentation of the project**

The goal of this project is to use and implement online photo sharing API platforms using React native, a JavaScript mobile application framework and [Imgur](https://apidocs.imgur.com/) API.

1. **Utilization**

To use the project, clone the repository then execute:

* yarn install
* Define the ANDROID\_SDK\_ROOT environment variable
* react-native run-android
* You must have a device connected to your computer or an emulator[[1]](#footnote-1) launched to be able
* to run the project. After those commands, the application would appear on your device.
* You can now use it.

1. **Functionalities**

* *Imgur* API implementation
* Authenticate to the *Imgur* platform
* Display the photos put online by the user connected
* Search for photos on the platform
* Upload photos to the platform
* Manage your favorites
* Filter the displayed photos

# **PRESENTATION OF THE PROJECT**

1. **Architecture**

I have divided the project into several different folders to make it clearer. Here is the architecture of the folders that can be found in src/.

**Api:** contains different requests from Imgur API.

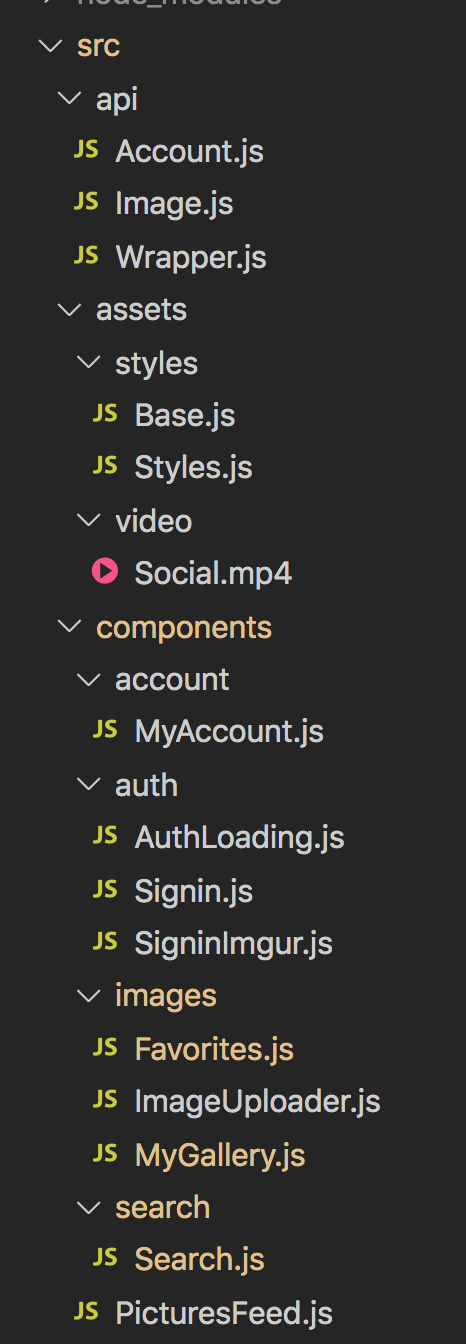
**Assets:** is divided into two folders:

* **styles** that contains a shared style sheet and a custom component (*BaseContain*er) used in the renders of others components
* **video** that contains the mp4 file used in the background of the login screen.

**Components:** contains the different components displayed in the user interface. You can refer to the name to understand the app page displayed. *(Feed file refer to the pictures feed of the App ~ home)*

Two others files are important too, there are in the base of the application:

* **Navigation.js**: the app navigation gestion *(StakNavigator and ButomTabNavigator)* integrated in the file:
* **App.js** that contains the base of the App.

****

1. **Dependencies**

* **[React-native-image -picker](https://github.com/react-native-community/react-native-image-picker)**

This dependency is used to handle the camera and the gallery, to upload pictures from the mobile device.

* **[React-native-picker-select](https://www.npmjs.com/package/react-native-picker-select)**

Is used to select items (filter in the pictures feed view and select the image to upload)

* **React-native-vector-icons**

To have responsive icons, i used the react-native-vector-icons dependency.

* **React-native-video**

I used the react-native-video to display a video in the background of the login screen.

* **React-navigation**

The react-navigation is used to handle the navigation between each view in the mobile application. I used the *‘StackNavigator’* and ‘*buttomTabNavigator’*.

* **[Gallery](https://gitlab.thinknet.co.th/jirawat/react-native-image-gallery)**

The Gallery dependency is used to display the images in the Pictures feed, My gallery, Favorites and Search views.

* **[Axios](https://github.com/axios/axios)**

I used the Axios dependency to handle the API request (= promise based HTTP client for the browser and node.js)

* **React-native webview**

To handle the connection with the Imgur’s API, i used a webview, displayed by the react-native- webview dependency.

* [**Jest**](https://jestjs.io/docs/en/tutorial-react-native) **and Enzyme**

These dependencies are used to handle tests of my project

1. **Automated tests**

In order to test my CRNA (*Create React Native Application*) and allow me to quickly verify that I’m not “breaking anything’ and that my new code is working properly**, I used Jest with Enzyme.**

**JEST**

Developed by Facebook, Jest is a JavaScript testing library for React web-applications. With Jest automated testing framework, i can perform snapshot testing to help me comprehend the state of my react trees which are transforming over time. It provides an excellent combination of a test runner & an assertion library.

**ENZYME**

**Enzyme** is used to facilitate the test and allow me to easily test my isolated components without executing the render() of the child component, which is practical and much faster.

**MOCK**

In order to be deterministic in the tests, all the external dependencies are mocked using Jest.

For the native components I let Jest do it automatically.

For the custom components I’m creating the mock by myself in order to have the control on the input/output

1. **Difference between an emulator and a simulator**

   **Emulator (ANDROID**) reproduces the necessary environment for running an Android application (Software and Hardware), available on Windows, Linux, Mac Os.

   **Simulator (IOS):** Only reproduces the software part, therefore, available on MAC Os (which has the hardware part). [↑](#footnote-ref-1)